SIAM J. SCI. COMPUT.
Vol. 20, No. 2, pp. 490–514

STABILIZING THE HIERARCHICAL BASIS BY APPROXIMATE WAVELETS II: IMPLEMENTATION AND NUMERICAL RESULTS*

PANAYOT S. VASSILEVSKI† AND JUNPING WANG‡

Abstract. This paper is the second part of a work on stabilizing the classical hierarchical basis HB by using wavelet-like basis functions. Implementation techniques are of major concern for the multilevel preconditioners proposed by the authors in the first part of the work, which deals with algorithms and their mathematical theory. Numerical results are presented to confirm the theory established there. A comparison of the performance of a number of multilevel methods is conducted for elliptic problems of three space variables.

Key words. hierarchical basis, multilevel methods, preconditioning, finite element elliptic equations, approximate wavelets

AMS subject classifications. 65F10, 65N20, 65N30

PII. S1064827596300668

1. Introduction.

In this paper we are concerned with implementation techniques on the wavelet-modified hierarchical basis method proposed in [16]. As discussed in [16], the method stabilizes the classical HB [18] by taking away from each HB function its approximate L^2-projections on coarse levels. The modified and stabilized hierarchical basis shall be called Approximate Wavelet-Modified Hierarchical Basis (AWM-HB).

The AWM-HB is viewed as a stabilization of the HB in the sense that it provides a stable Riesz basis in the Sobolev space $H^\alpha(\Omega)$ for $\alpha \in (0, 1]$. As a result, it can be employed to yield optimal preconditioners for finite element discretizations of elliptic problems.

Other stabilizations of the HB methods, such as the AMLI method presented in [2] and [14], are not of V-cycle type, whereas the AWM-HB is. The multiplicative AWM-HB method fits in the general framework as given in Vassilevski [13] and [14], which is an extension of the two-level method proposed by Bank and Dupont [3] and studied further by Axelsson and Gustafsson [1].

A survey on the subject of HB stabilization can be found in Vassilevski [15]. Other related results in the use of L^2 and H^1 orthogonal direct decompositions for finite element spaces can be found in Griebel and Oswald [9] and Stevenson [11], [12]. Similar constructions for wavelets were exploited in Carnicer, Dahmen, and Peña [7]. Our result is general in that it is of optimal order and applies to cases wherever the standard finite element HB decomposition exists.

To implement the proposed AWM-HB preconditioners, we reformulate the algorithm of [16] in a matrix form. Computationally feasible algorithms are also designed to compute the action of the approximate L^2-projection operator Q^a_{k-1} on functions

*Received by the editors March 15, 1996; accepted for publication (in revised form) May 12, 1997; published electronically September 10, 1998. The work of the first author was partially supported by the U.S. NSF grant INT-95-06184 and by Bulgarian Ministry for Education, Science, and Technology under grant I–504, 1995. The second author was partially supported by NSF grant INT-93-09286.
http://www.siam.org/journals/sisc/20-2/30066.html

†Center of Informatics and Computer Technology, Bulgarian Academy of Sciences, Acad. G. Bontchev street, Block 25A, 1113 Sofia, Bulgaria (panayot@iscbg.acad.bg).

‡Department of Mathematics, University of Wyoming, P.O. Box 3036, Laramie, WY 82071-3036 (wang@schwarz.uwyo.edu).

$v \in V_k^{(1)}$, where k indicates the number of levels, with larger k corresponding to finer spaces. Details can be found in sections 2.2 and 4.

The major concern of this paper is to report some numerical results on the performance of various preconditioners involving HB, AWM-HB, and multigrid (MG) methods for elliptic problems in three dimensions (3D). Our comparison shows a superior performance in the CPU timing of the MG method over other tested algorithms. The performance of AWM-HB falls in between HB and MG. Thus, AWM-HB can be used as a stabilization technique to any available 3D HB code, especially for elliptic problems discretized on highly nonuniform grids with local refinements.

The paper is organized as follows. In section 2, we review the additive and multiplicative preconditioners arising from the AWM-HB method. In section 3, we present some examples on the approximate L^2-projection. In section 4, we reformulate the AWM-HB preconditioners in a matrix-vector form. In section 5.1, we present some numerical results which illustrate the theory developed in [16]. Finally, in section 5.2 we present a comparison test on various preconditioning methods for problems in 3D.

2. Preliminaries.

2.1. A model problem and its discretization. The bilinear form under consideration is given as follows:

$$a(\varphi, \psi) = \int_\Omega a \nabla\varphi \cdot \nabla\psi \qquad \forall \varphi,\ \psi \in H_0^1(\Omega). \tag{2.1}$$

Here $a = \{a_{ij}(x)\}$ is a coefficient matrix, which is assumed to be symmetric and positive definite uniformly in $x \in \Omega$ with bounded and measurable entries $a_{ij}(x)$.

To discretize the bilinear form $a(\cdot, \cdot)$, we use the routine successive (possibly local) refinement procedure to generate a sequence of finite element triangulations $\mathcal{T}_k$ for $k = 0, 1, \ldots, J$, with $\mathcal{T}_0$ being the initial triangulation. Let V_k be the conforming piecewise-linear finite element space associated with $\mathcal{T}_k$. Denote by $A^{(k)} : V_k \to V_k$ the corresponding discretization of the bilinear form given by

$$(A^{(k)}\varphi, \psi) = a(\varphi, \psi) \qquad \forall \varphi, \psi \in V_k,$$

where $(\cdot, \cdot)$ stands for the standard L^2-inner product.

Each V_k is equipped with a standard Lagrangian (nodal) basis $\{\phi_i^{(k)},\ x_i \in \mathcal{N}_k\}$, where $\mathcal{N}_k$ is the node set (the set of nodal degrees of freedom) of V_k. The basis functions satisfy $\phi_i^{(k)}(x_j) = \delta_{i,j}$—the Kronecker symbol when x_j runs over the node set $\mathcal{N}_k$. We assume that $\mathcal{N}_k \subset \mathcal{N}_{k+1}$. Due to the refinement process we have $V_k \subset V_{k+1}$.

Let Q_k be the L^2-projection operator from $L^2(\Omega)$ to V_k defined by

$$(Q_k v, \psi) = (v, \psi) \quad \forall \psi \in V_k.$$

It is clear that the action $Q_k v$ requires inverting a mass (or Gram) matrix. Let $I_k :\ C(\bar{\Omega}) \to V_k$ be a nodal interpolation operator given as follows:

$$I_k v = \sum_{x_i \in \mathcal{N}_k} v(x_i)\phi_i^{(k)}.$$

Finally, let Q_k^a be an approximation of Q_k satisfying

$$\|(Q_k - Q_k^a)v\|_0 \le \tau \|Q_k v\|_0 \quad \forall v \in L^2(\Omega) \tag{2.2}$$

for a prescribed small tolerance $\tau \ge 0$.

2.2. The AWM-HB preconditioners. The AWM-HB preconditioners exploit the following direct decomposition for each V_k:

$$V_k = V_k^1 \oplus V_{k-1},$$

where $V_k^1 = (I - Q_{k-1}^a)V_k^{(1)}$ and $V_k^{(1)} = (I_k - I_{k-1})V_k$. Using the above decomposition recursively we obtain the following:

$$V = V_0 \oplus V_1^1 \oplus V_2^1 \oplus \cdots \oplus V_J^1.$$

Let $\mathcal{N}_k^{(1)} = \mathcal{N}_k \setminus \mathcal{N}_{k-1}$. It is not hard to see that the set of functions

$$\{(I - Q_{k-1}^a)\phi_i^{(k)} : \ x_i \in \mathcal{N}_k^{(1)}\} \tag{2.3}$$

forms a basis of V_k^1. The new basis $\{(I - Q_{k-1}^a)\phi_i^{(k)}\}$ is clearly a modification of the classical HB functions of $V_k^{(1)}$; the modification was made by taking away from the HB function $\phi_i^{(k)}$ its approximate L^2-projection onto the nearest coarse space V_{k-1}.

The following operators are needed in the construction of the AWM-HB preconditioners:

- The solution operator $A_{11}^{(k)} : \ V_k^1 \to V_k^1$ as the restriction of $A^{(k)}$ onto the subspace V_k^1. $A_{11}^{(k)}$ is defined as follows:

$$(A_{11}^{(k)}\psi^1, \varphi^1) = a(\psi^1, \varphi^1) \qquad \forall \varphi^1, \ \psi^1 \in V_k^1. \tag{2.4}$$

- $A_{12}^{(k)} : \ V_{k-1} \to V_k^1$ and $A_{21}^{(k)} : \ V_k^1 \to V_{k-1}$ are given by

$$(A_{12}^{(k)}\widetilde{\psi}, \varphi^1) = (\widetilde{\psi}, A_{21}^{(k)}\varphi^1) = a(\varphi^1, \widetilde{\psi}) \qquad \forall \widetilde{\psi} \in V_{k-1}, \varphi^1 \in V_k^1. \tag{2.5}$$

With the above notation, the operator $A^{(k)}$ naturally admits the following two-by-two block decomposition:

$$A^{(k)} = \begin{bmatrix} A_{11}^{(k)} & A_{12}^{(k)} \\ A_{21}^{(k)} & A^{(k-1)} \end{bmatrix} \begin{matrix} \} & V_k^1 \\ \} & V_{k-1} \end{matrix} . \tag{2.6}$$

Let $B_{11}^{(k)}$ be given symmetric and positive definite matrices which are spectrally equivalent to $A_{11}^{(k)}$:

$$(A_{11}^{(k)}\varphi^1, \varphi^1) \le (B_{11}^{(k)}\varphi^1, \varphi^1) \le (1 + b_1)(A_{11}^{(k)}\varphi^1, \varphi^1) \qquad \forall \varphi^1 \in V_k^1. \tag{2.7}$$

Here b_1 is an absolute constant.

Let $A = A^{(J)}$ be the operator of major concern. Below, we define two preconditioners B and D that exploit the two-by-two block structure of each $A^{(k)}$ in (2.6).

DEFINITION 2.1 (multiplicative AWM-HB preconditioners). *The multiplicative AWM-HB preconditioner of $A = A^{(J)}$, denoted by $B = B^{(J)}$, is defined by the following procedure:*

- *Set $B^{(0)} = A^{(0)}$.*
- *For $k = 1, \ldots, J$, set*

$$B^{(k)} = \begin{bmatrix} B_{11}^{(k)} & 0 \\ A_{21}^{(k)} & B^{(k-1)} \end{bmatrix} \begin{bmatrix} I & B_{11}^{(k)^{-1}} A_{12}^{(k)} \\ 0 & I \end{bmatrix} \begin{matrix} \} & V_k^1 \\ \} & V_{k-1} \end{matrix} .$$

DEFINITION 2.2 (additive AWM-HB preconditioners). *The additive AWM-HB preconditioner of $A = A^{(J)}$, denoted by $D = D^{(J)}$, is defined by the following procedure:*

- *Set $D^{(0)} = A^{(0)}$.*
- *For $k = 1, \ldots, J$, set*

$$D^{(k)} = \begin{bmatrix} B_{11}^{(k)} & 0 \\ 0 & D^{(k-1)} \end{bmatrix} \begin{matrix} \} & V_k^1 \\ \} & V_{k-1} \end{matrix} .$$

2.3. Main results for the AWM-HB preconditioners. In [16], we have established a spectral equivalence between A and its preconditioners B and D. More precisely, the following result was derived:

$$c_1(Sv, v) \le (Av, v) \le c_2(Sv, v) \qquad \forall v \in V_J, \tag{2.8}$$

where $S = B^{(J)}$ or $D^{(J)}$. Here c_i are absolute constants independent of the mesh size h. The estimate (2.8) is based on the following assumptions:

(A) The tolerance τ in (2.2) must be sufficiently small, but independent of the mesh sizes h_i or the level number J. More precisely, if C_R is chosen such that

$$\|(I_k - I_{k-1})v\|_0 \le C_R \|v\|_0 \quad \forall v \in V_k$$

and $h_k = \frac{1}{2} h_{k-1}$ for the mesh size h_k, then τ is determined by

$$\tau C_R \le q < 1 \tag{2.9}$$

for any fixed constant q.

(B) There exists a constant $\sigma_N > 0$ such that the following estimate holds:

$$|Q_0 v|_1^2 + \sum_{s=1}^{J} 2^{2s} \|(Q_s - Q_{s-1})v\|_0^2 \le \sigma_N \|v\|_1^2 \qquad \forall v \in V.$$

(C) There exist constants $\sigma_I > 0$ and $\delta \in (0, 1)$ (in fact, if $h_i = \frac{1}{2} h_{i-1}$, then $\delta = \frac{1}{\sqrt{2}}$) such that the following strengthened Cauchy–Schwarz inequality holds for any $i \le j$:

$$a(v, w)^2 \le \sigma_I \delta^{2(j-i)} a(v, v) \lambda_j \|w\|_0^2 \qquad \forall v \in V_i,\ w \in V_j.$$

Here $\lambda_j = O(h_j^{-2})$ is the largest eigenvalue of the operator $A^{(j)}$.

The assumptions (B) and (C) have been respectively verified by Oswald [10] and Yserentant [18, 19]; see also [17], [5], and [8]. Note that (B) and (C) are the minimal assumptions used in the modern convergence theory of the classical MG method.

The assumption (A) can be verified easily for quasi-regular partitions of the domain Ω. In fact, the standard L^2-inner product is equivalent to the following discrete version:

$$(v, w)_{0,\,k} \equiv h_k^d \sum_{x_i \in \mathcal{N}_k} v(x_i) w(x_i), \quad v,\ w \in V_k$$

in the finite element space V_k. More precisely, there are absolute positive constants γ_1 and γ_2 such that

$$\gamma_1 \|v\|_0^2 \le (v, v)_{0,\,k} \le \gamma_2 \|v\|_0^2 \quad \forall v \in V_k.$$

It follows that

$$\|(I_k - I_{k-1})v\|_0^2 \le \gamma_1^{-1} h_k^d \sum_{x_i \in \mathcal{N}_k \setminus \mathcal{N}_{k-1}} v^2(x_i) \le \frac{\gamma_2}{\gamma_1} \|v\|_0^2.$$

This shows that $C_R^2 \le \mathrm{cond}\left(D_k^{-1} G_k\right)$, where G_k is the mass (Gram) matrix at level k and D_k is its diagonal. It can be seen that $\mathrm{cond}\left(D_k^{-1} G_k\right)$ is uniformly bounded from above by a positive number $\kappa \le \frac{\gamma_2}{\gamma_1}$. Consequently, it suffices to choose τ such that $\tau < \kappa^{-\frac{1}{2}}$. For the tetrahedra elements to be discussed in section 5.2 we have $\kappa = 5$. The corresponding estimate on triangular elements is $\kappa = 4$.

One of the important features in the decomposition (2.6) is that the block $A_{11}^{(k)}$ is well conditioned. In particular, it is spectrally equivalent to the diagonal part in its matrix representation with respect to the AWM-HB. Thus, the Richardson preconditioner would be a good choice for $B_{11}^{(k)}$ in approximating $A_{11}^{(k)}$ (as in (2.7)).

3. On the approximate L^2-projection. Let Q_{k-1}^a denote any approximate L^2-projection onto the subspace V_{k-1}. In practical computation, the operator Q_{k-1}^a is constructed by approximating the solution of

$$(Q_{k-1}v, w) = (v, w) \quad \forall\ w \in V_{k-1} \tag{3.1}$$

by simple iterative methods such as Jacobi or Gauss–Seidel iterations.

We now describe algorithms for computing the actions of Q_{k-1}^a. For any $v \in V_k^{(1)}$, let $\mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ 0 \end{bmatrix} \begin{matrix} \}\mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \}\mathcal{N}_{k-1} \end{matrix}$ be its coefficient vector with respect to the standard nodal basis of V_k; the second block component of $\mathbf{v}$ is zero since v vanishes on $\mathcal{N}_{k-1}$. Let $I_{k-1}^k = \begin{bmatrix} J_{12} \\ I \end{bmatrix} \begin{matrix} \}\mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \}\mathcal{N}_{k-1} \end{matrix}$ (with the abbreviation $J_{12} = J_{12}^{(k)}$) and $I_k^{k-1} = I_{k-1}^{k^T}$ be the natural coarse-to-fine, and respectively, fine-to-coarse transformation matrices. For example, if the nodal basis coefficient vector of a function $v_2 \in V_{k-1}$ in terms of the nodal basis of V_{k-1} is $\mathbf{v}_2$, then its coefficient vector with respect to the nodal basis of V_k (note that $v_2 \in V_{k-1} \subset V_k$) will be $I_{k-1}^k \mathbf{v}_2 = \begin{bmatrix} J_{12}\mathbf{v}_2 \\ \mathbf{v}_2 \end{bmatrix} \begin{matrix} \}\mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \}\mathcal{N}_{k-1}. \end{matrix}$

The action of J_{12} on vectors $\mathbf{v}_2$ can be carried out as in the following algorithm (cf. [18]).

ALGORITHM 3.1 (computing actions of J_{12}). *Let $v_2 \in V_{k-1}$ be the piecewise linear function corresponding to the vector $\mathbf{v}_2$ in the standard nodal basis of V_{k-1}. The entries of $\mathbf{v}_1 = J_{12}^{(k)} \mathbf{v}_2$ are the values of the function v_2 at the nodes $x_i \in \mathcal{N}_k \setminus \mathcal{N}_{k-1}$. More precisely, for each $x_i \in \mathcal{N}_k \setminus \mathcal{N}_{k-1}$, the corresponding component is given by*

$$v(x_i) = \frac{1}{2}\left(v_2(x_{i_1}) + v_2(x_{i_2})\right).$$

Here, x_{i_1} and $x_{i_2} \in \mathcal{N}_{k-1}$ are the endpoints of an edge E on the $(k-1)$th level such that $x_i \in \mathcal{N}_k \setminus \mathcal{N}_{k-1}$ is the midpoint of E.

Denote now by $G_k = \{(\phi_j^{(k)}, \phi_i^{(k)})\}_{x_j, x_i \in \mathcal{N}_k}$ the mass matrix of level k. Then (3.1) admits the following matrix-vector form:

$$\mathbf{w}_2^T G_{k-1} \mathbf{v}_2 = (I_{k-1}^k \mathbf{w}_2)^T G_k \mathbf{v} \qquad \forall\ \mathbf{w}_2.$$

Here $\mathbf{v}_2$ and $\mathbf{w}_2$ are, respectively, the nodal coefficient vectors of $Q_{k-1}v$ and $w \in V_{k-1}$. Therefore, we only need to solve the following mass-matrix problem:

$$G_{k-1}\mathbf{v}_2 = I_k^{k-1} G_k \mathbf{v}. \tag{3.2}$$

In other words, the exact L^2-projection $Q_{k-1}v$ is actually given by $G_{k-1}^{-1}I_k^{k-1}G_k\mathbf{v}$. Hence,

$$\|Q_{k-1}v\|_0^2 = \left(G_{k-1}^{-1}I_k^{k-1}G_k\mathbf{v}\right)^T G_{k-1}\left(G_{k-1}^{-1}I_k^{k-1}G_k\mathbf{v}\right) = \|G_{k-1}^{-\frac{1}{2}}I_k^{k-1}G_k\mathbf{v}\|^2. \tag{3.3}$$

Here and in what follows we use the notation $\|\mathbf{x}\|^2 = \mathbf{x}^T\mathbf{x}$.

To have a computationally feasible basis, we have to replace G_{k-1}^{-1} by some approximations $\widetilde{G}_{k-1}^{-1}$ whose action can be computed by simple iterative methods applied to (3.2). One possibility is to use any classical splitting of G_{k-1} and consider approximations of the form $\widetilde{G}_{k-1} = (D_{k-1} - L_{k-1})D_{k-1}^{-1}(D_{k-1} - U_{k-1})$, where D_{k-1}, L_{k-1}, and U_{k-1} are, respectively, diagonal, lower-triangular, and upper-triangular sparse matrices. The symmetrized Gauss–Seidel method is employed to approximate G_{k-1}^{-1} in our numerical experiments.

Other approximations are also feasible in the computation. For example, we may evaluate the left-hand side of (3.1) by using simple quadrature rules, leading to invertible approximations $\widetilde{G}_{k-1}$ for the mass matrix G_{k-1}. In particular, if the rule

$$\int_T \psi \, dx \approx \frac{|T|}{n_v}\sum_{i=1}^{n_v}\psi(x_i)$$

is employed on each element T (with $|T|$ the area and $\{x_i\}_{i=1}^{n_v}$ the set of vertices of T), then the resulting approximation $\widetilde{G}_{k-1}$ would be a diagonal matrix. We point out that this approximation may not be as accurate as required by (2.2), though a spectral equivalence can be easily seen by using an element-based local analysis.

A good remedy for the above drawback is the following. Let B_{k-1} be an approximation of G_{k-1}. If the required accuracy (2.2) is not achieved, we would consider the following polynomial approximation to G_{k-1}^{-1}:

$$\widetilde{G}_{k-1}^{-1} = \left[I - \pi_m\left(B_{k-1}^{-1}G_{k-1}\right)\right]G_{k-1}^{-1}, \tag{3.4}$$

where π_m is a polynomial of degree $m \geq 1$. The polynomial π_m also satisfies $\pi_m(0) = 1$ and $0 \leq \pi_m(t) < 1$ for $t \in [\alpha, \beta]$, where the latter interval contains the spectrum of the scaled mass matrix $B_{k-1}^{-1}G_{k-1}$. Since $B_{k-1}^{-1}G_{k-1}$ is well conditioned, the interval $[\alpha, \beta]$ can be chosen to be independent of k. Thus, there is a fixed polynomial π_m of degree m such that the resulting approximation $\widetilde{G}_{k-1}^{-1}$ satisfies the required accuracy in (2.2).

Now comes the estimate between Q_{k-1}^a and Q_{k-1}. If Q_{k-1}^a is given by $\widetilde{G}_{k-1}^{-1}I_k^{k-1}G_k$ and (3.4), we have

$$\begin{aligned}
\|Q_{k-1}^a v - Q_{k-1}v\|_0 &= \left\|G_{k-1}^{\frac{1}{2}}\left(G_{k-1}^{-1} - \widetilde{G}_{k-1}^{-1}\right)I_k^{k-1}G_k\mathbf{v}\right\| \\
&= \left\|G_{k-1}^{\frac{1}{2}}\pi_m\left(B_{k-1}^{-1}G_{k-1}\right)G_{k-1}^{-1}I_k^{k-1}G_k\mathbf{v}\right\| \\
&= \left\|\pi_m\left(G_{k-1}^{\frac{1}{2}}B_{k-1}^{-1}G_{k-1}^{\frac{1}{2}}\right)G_{k-1}^{-\frac{1}{2}}I_k^{k-1}G_k\mathbf{v}\right\| \\
&\leq \max_{t\in[\alpha,\beta]}\pi_m(t)\left\|G_{k-1}^{-\frac{1}{2}}I_k^{k-1}G_k\mathbf{v}\right\| \\
&= \max_{t\in[\alpha,\beta]}\pi_m(t)\|Q_{k-1}v\|_0.
\end{aligned}$$

Here we have used the identity (3.3) and the properties of π_m. Thus, the polynomial π_m should be selected so that

$$\tau \geq \max_{t\in[\alpha,\beta]} \pi_m(t)$$

for a prescribed small parameter τ.

A simple choice of $\pi_m(t)$ is the truncated series

$$(1-\pi_m(t))t^{-1} = p_{m-1}(t) \equiv \beta^{-1}\sum_{k=0}^{m-1}\left(1-\frac{1}{\beta}t\right)^k, \tag{3.5}$$

which yields $\widetilde{G}_{k-1}^{-1} = p_{m-1}(B_{k-1}^{-1}G_{k-1})B_{k-1}^{-1}$. We remark that (3.5) was obtained from the following expansion:

$$1 = t\beta^{-1}\sum_{k=0}^{\infty}(1-t\beta^{-1})^k, \qquad t\in[\alpha,\beta].$$

With the above choice on the polynomial $\pi_m(t)$, we have

$$\pi_m(t) = 1 - tp_{m-1}(t) = t\beta^{-1}\sum_{k\geq m}(1-\beta^{-1}t)^k = (1-\beta^{-1}t)^m.$$

It follows that

$$\max_{t\in[\alpha,\beta]}\ \pi_m(t) = \left(1-\frac{\alpha}{\beta}\right)^m.$$

The best choice of π_m, as is well known, is given by the Chebyshev polynomial. In this case, we have $\max_{t\in[\alpha,\beta]}\pi_m(t) \leq 2\delta^m/(1+\delta^{2m}) < 2\delta^m$, where

$$\delta = (\sqrt{\kappa}-1)/(\sqrt{\kappa}+1) < 1$$

and $\kappa = \beta/\alpha$ is independent of k.

In our numerical experiments, we will be using $m \geq 1$ steps of some stationary iterative method applied to (3.2) with a convergence factor $\rho < 1$. The restriction on $\tau < C_R^{-1}$ (see assumption (A)) is then translated to $\rho^m < C_R^{-1}$. It follows that

$$m > \log(C_R)/\log(\rho^{-1}). \tag{3.6}$$

For example, if the Jacobi method is employed to approximate the mass matrix, then the diagonal part $B_{k-1} = D_{k-1}$ of G_{k-1} is actually the preconditioner and the following estimate is valid:

- for triangular piecewise-linear elements,

$$\frac{1}{2}\mathbf{v}^T D_{k-1}\mathbf{v} \leq \mathbf{v}^T G_{k-1}\mathbf{v} \leq 2\mathbf{v}^T D_{k-1}\mathbf{v} \quad \forall\ \mathbf{v};$$

- for tetrahedral piecewise-linear elements,

$$\frac{1}{2}\mathbf{v}^T D_{k-1}\mathbf{v} \leq \mathbf{v}^T G_{k-1}\mathbf{v} \leq \frac{5}{2}\mathbf{v}^T D_{k-1}\mathbf{v} \quad \forall\ \mathbf{v}.$$

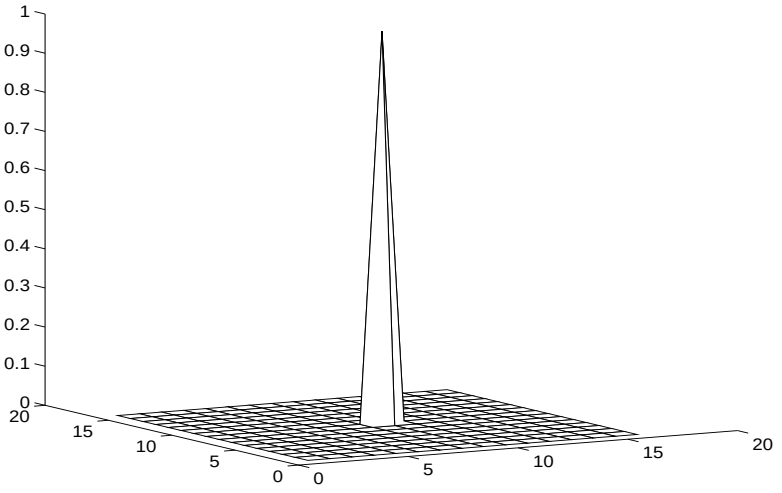

FIG. 1. *Plot of hierarchical basis functions.*

Thus, with the optimal choice of the parameter $\omega = \frac{2}{\gamma_1+\gamma_2}$ in the iterative matrix $I - \omega D_{k-1}^{-1} G_{k-1}$, the convergence factor for triangular elements (for which $\gamma_1 = \frac{1}{2}$, $\gamma_2 = 2$) is bounded by $\rho = 3/5$. Using (3.6) and the fact that

$$C_R \leq \operatorname{cond}\left(D_{k-1}^{-1} G_{k-1}\right)^{\frac{1}{2}} \leq 2,$$

we arrive at the following estimate:

$$m > \log 2/(\log 5 - \log 3) = 1.3569.$$

In other words, it suffices to perform $m = 2$ scaled Jacobi iterations in order to reach the required accuracy for the approximate L^2 projections.

For tetrahedral elements, we have $C_R \leq \sqrt{5}$, $\gamma_1 = \frac{1}{2}$, and $\gamma_2 = \frac{5}{2}$. Then the Jacobi method converges with rate bounded by $\rho = 2/3$. The estimate (3.6) leads to $2m > \log 5/(\log 3 - \log 2) = 3.9694$. Again, it suffices to perform $m = 2$ Jacobi iterations in order to obtain an H^1-stable basis.

In addition to the Jacobi approximation, we have also used the symmetric Gauss–Seidel approximation to the mass matrix in our numerical experiments. This method is more accurate than the Jacobi.

In conclusion, the requirement (3.6) imposes a very mild restriction on m. In practice, we expect to have small m (say, $m = 1, 2$) for any reasonably good approximations B_{k-1}. This observation is confirmed by our numerical experiments to be presented in section 5: $m = 2, 4$ for Jacobi approximations and $m = 1, 2$ for symmetric Gauss–Seidel approximations to G_{k-1}. We show in Fig. 1 a typical plot of a nodal basis function of $V_k^{(1)}$ and its approximate wavelet modifications resulting from the Jacobi method with $m = 1$ in Fig. 2, $m = 2$ in Fig. 3, and $m = 4$ in Fig. 4. Increasing m does not make much noticeable difference in the plot. We also show the corresponding approximate wavelet modification for B_{k-1} being the symmetric Gauss–Seidel approximation to G_{k-1} for $m = 1$ in Fig. 5 and $m = 2$ in Fig. 6. Note that this does not give locally supported AWM-HB functions, since $B_{k-1}^{-1} G_{k-1}$ is not a sparse matrix. In Fig. 7 we show the exact wavelet-modified HB function.

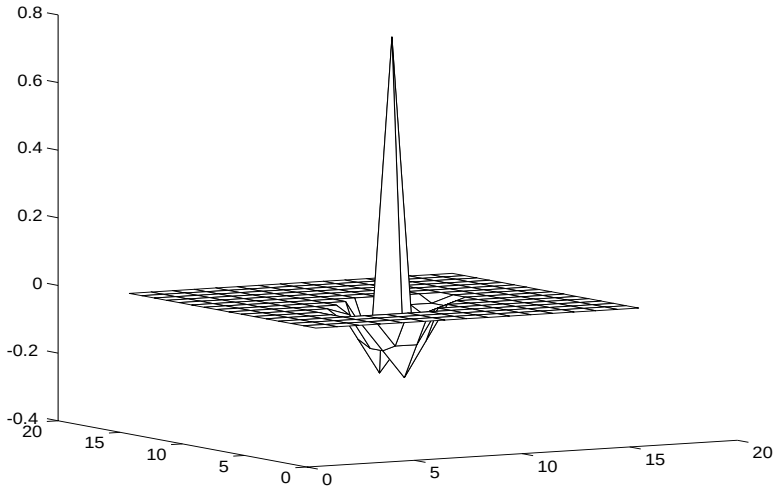

FIG. 2. *Plot of a wavelet-modified HB function; one* $(m = 1)$ *Jacobi iteration.*

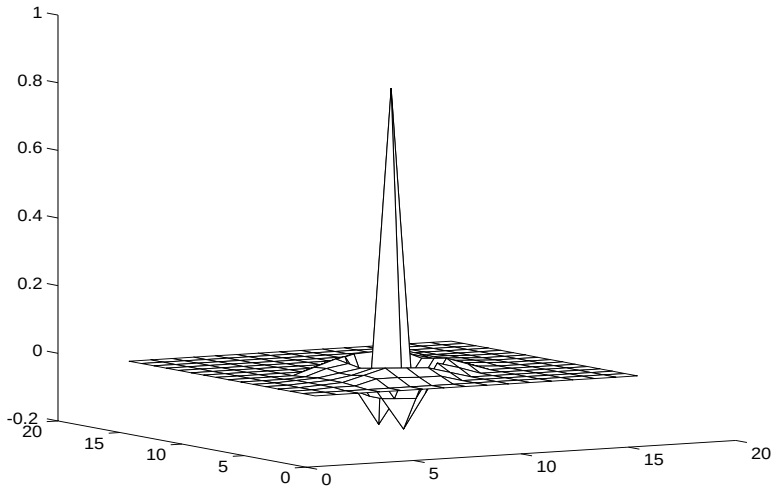

FIG. 3. *Plot of a wavelet-modified HB function; two* $(m = 2)$ *Jacobi iterations.*

It is seen that there is no visible difference among the graphs of the functions in Fig. 4, Fig. 6, and Fig. 7. This is due to the exponential decay property of the exact wavelet-modified HB function. The cross-section plot of various approximations is shown in Figs. 8–13. The conjugate gradient method with 16 iterations was employed to provide the "exact" solution of the mass-matrix problem (3.2) for the plots in Figs. 7 and 13.

4. Matrix representations of the AWM-HB methods. We now turn to a description of the multiplicative and additive AWM-HB methods in matrix forms.

Let us first derive matrix representations for the operators $A_{11}^{(k)}$, $A_{12}^{(k)}$, and $A_{21}^{(k)}$ introduced in section 2. In what follows of this section, capital letters without hats will denote matrices corresponding to the standard nodal basis of the underlined finite

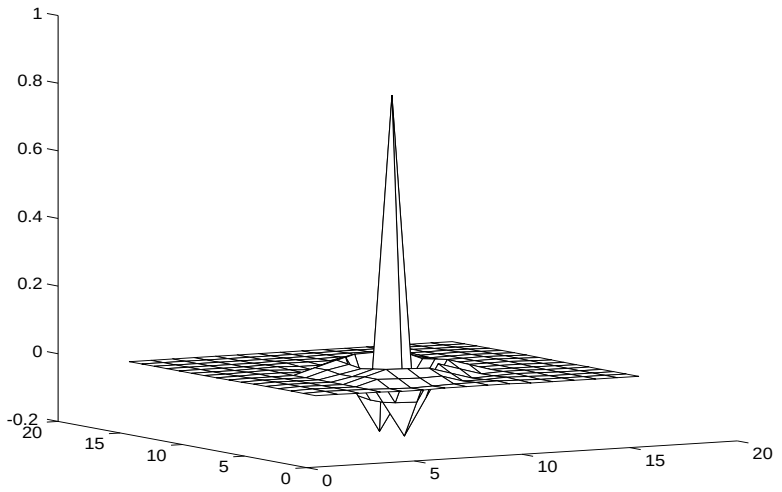

FIG. 4. *Plot of a wavelet-modified HB function; four* ($m = 4$) *Jacobi iterations.*

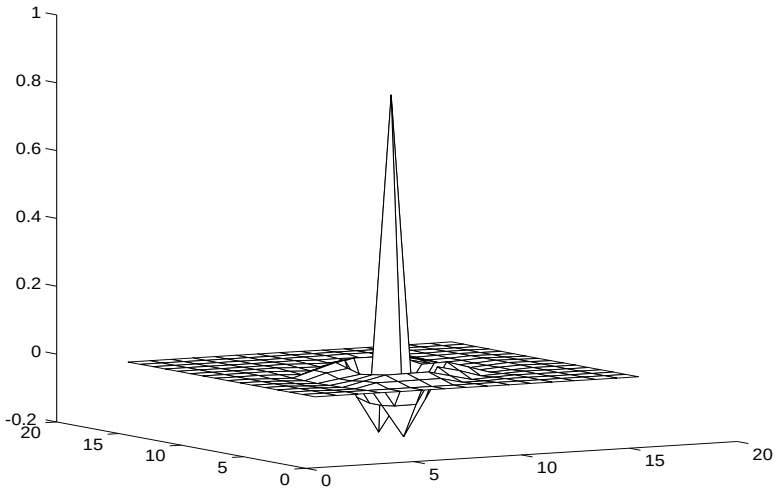

FIG. 5. *Plot of a wavelet-modified HB function; one* ($m = 1$) *Gauss–Seidel iteration.*

element space. For example, $A^{(k)}$ denotes the standard nodal basis stiffness matrix with entries $\{a(\phi_i^{(k)}, \phi_j^{(k)})\}_{x_i, x_j \in \mathcal{N}_k}$.

For any $v \in V_k$ and its nodal coefficient vector $\mathbf{v}$, we decompose v as follows:

$$v = (I - Q_{k-1}^a)(I_k - I_{k-1})v + w_2,$$

where $w_2 \in V_{k-1}$ is uniquely determined as $w_2 = I_{k-1}v + Q_{k-1}^a(I_k - I_{k-1})v$. Our goal is to find a vector representation for the components of v. Since the above decomposition is direct, it is clear that there are vectors $\widehat{\mathbf{v}}_1$ and $\widehat{\mathbf{v}}_2$ satisfying

$$\mathbf{v} = Y_1^{(k)}\widehat{\mathbf{v}}_1 + Y_2^{(k)}\widehat{\mathbf{v}}_2, \tag{4.1}$$

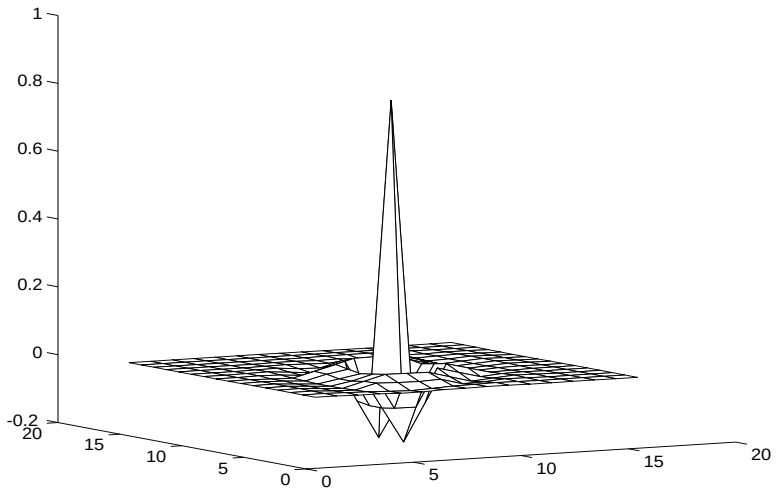

FIG. 6. *Plot of a wavelet-modified HB function; two* ($m = 2$) *Gauss–Seidel iterations.*

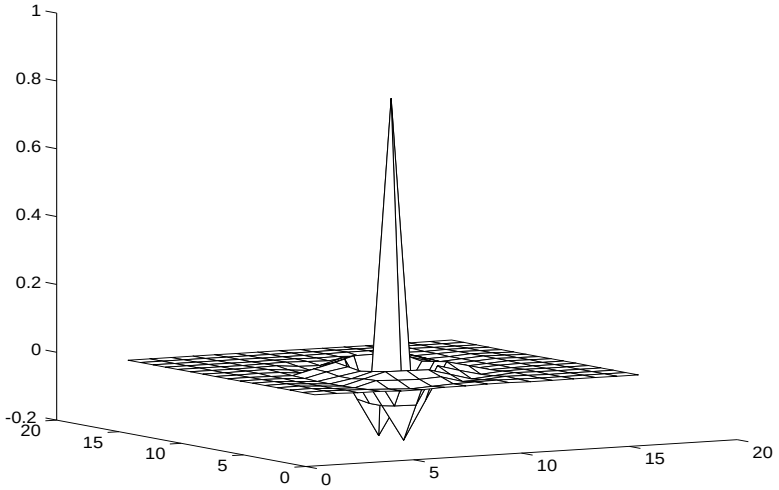

FIG. 7. *Plot of a wavelet-modified HB function; Sixteen CG iterations.*

where (to be shown below),

$$
\begin{aligned}
Y_1^{(k)} &= (I - I_{k-1}^k \widetilde{G}_{k-1}^{-1} I_k^{k-1} G_k) \begin{bmatrix} I \\ 0 \end{bmatrix} \begin{matrix} \}\mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \}\mathcal{N}_{k-1} \end{matrix}, \\
Y_2^{(k)} &= I_{k-1}^k = \begin{bmatrix} J_{12}^{(k)} \\ I \end{bmatrix} \begin{matrix} \}\mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \}\mathcal{N}_{k-1} \end{matrix}.
\end{aligned} \tag{4.2}
$$

The vectors $\widehat{\mathbf{v}}_1$ and $\widehat{\mathbf{v}}_2$ represent the two components of our wavelet-modified two–level HB coefficient vector $\widehat{\mathbf{v}} = \left[\begin{smallmatrix} \widehat{\mathbf{v}}_1 \\ \widehat{\mathbf{v}}_2 \end{smallmatrix}\right]$ of v.

Let $Y = [Y_1,\ Y_2]$, $Y_1 = Y_1^{(k)}$, and $Y_2 = Y_2^{(k)}$. We describe how $\widehat{\mathbf{v}} = Y^{-1}\mathbf{v}$ can be computed. In fact, we have

$$v = (I_k - I_{k-1})v + v_2, \qquad v_2 = I_{k-1}v.$$

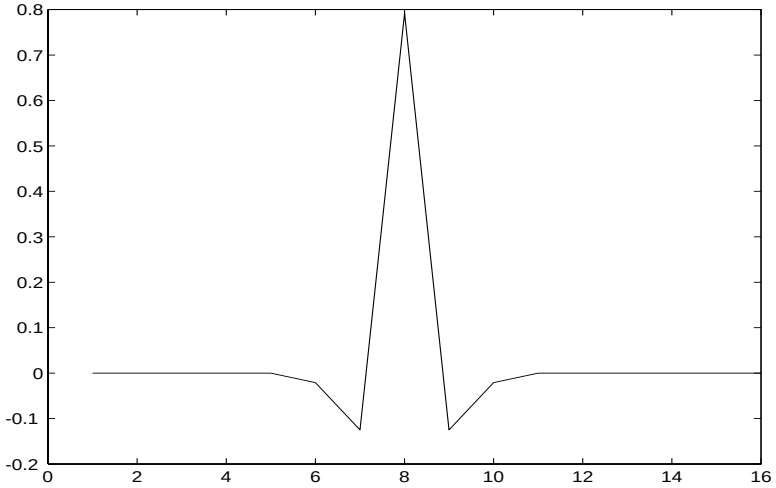

FIG. 8. *Cross-section plot of a wavelet-modified HB function; one Jacobi iteration.*

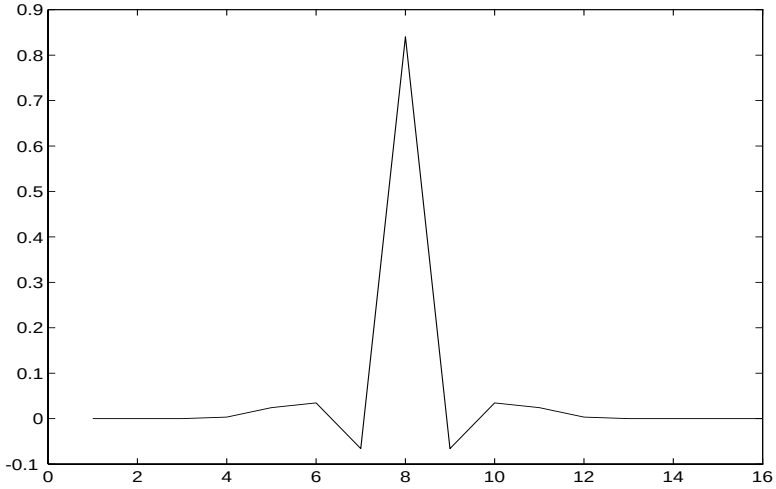

FIG. 9. *Cross-section plot of a wavelet-modified HB function; two Jacobi iterations.*

Let $\mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} \begin{matrix} \}\mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \}\mathcal{N}_{k-1} \end{matrix}$ be the standard nodal coefficient vector of $v \in V_k$. Then, the nodal coefficient vector of $(I_k - I_{k-1})v$ has the form

$$\mathbf{v} - I_{k-1}^k \mathbf{v}_2 = \begin{bmatrix} \mathbf{v}_1 - J_{12}^{(k)} \mathbf{v}_2 \\ 0 \end{bmatrix} \begin{matrix} \}\mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \}\mathcal{N}_{k-1} \end{matrix} ;$$

the second component is zero since $(I_k - I_{k-1})v$ vanishes on $\mathcal{N}_{k-1}$. Denote

$$\widehat{\mathbf{v}}_1 = \mathbf{v}_1 - J_{12}^{(k)} \mathbf{v}_2. \tag{4.3}$$

Then $\widehat{\mathbf{v}}_1$ is also the AWM-HB coefficient vector of $(I - Q_{k-1}^a)(I_k - I_{k-1})v$ expanded in terms of the AWM-HB functions $\{(I - Q_{k-1}^a)\phi_i^{(k)},\ x_i \in \mathcal{N}_k \setminus \mathcal{N}_{k-1}\}$ (which form the basis of V_k^1). Consequently, the second component $\widehat{\mathbf{v}}_2$, which is the coefficient vector

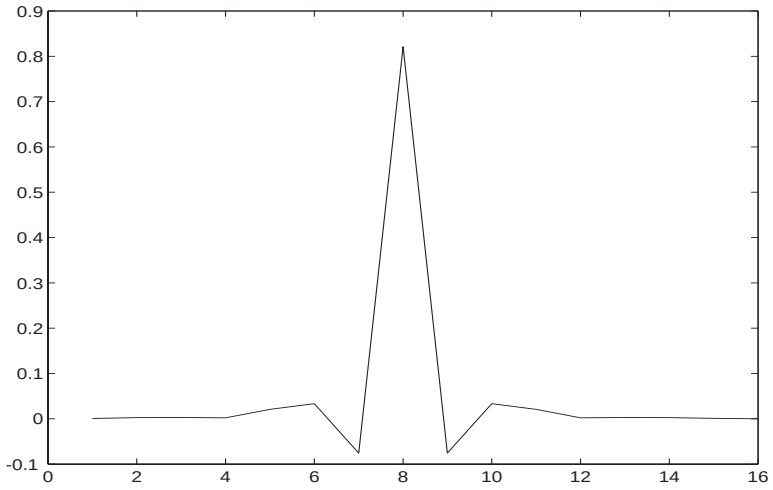

FIG. 10. *Cross-section plot of a wavelet-modified HB function; four Jacobi iterations.*

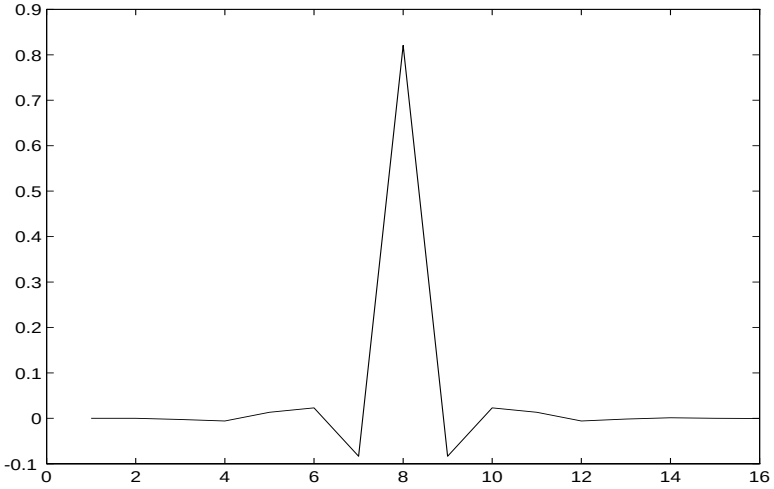

FIG. 11. *Cross-section plot of a wavelet-modified HB function; one Gauss–Seidel iteration.*

of $w_2 = I_{k-1}v + Q^a_{k-1}(I_k - I_{k-1})v \in V_{k-1}$ with respect to the $(k-1)$th level nodal basis, is given by

$$\begin{aligned} \widehat{\mathbf{v}}_2 &= \widetilde{G}^{-1}_{k-1} I^{k-1}_k G_k \begin{bmatrix} I \\ 0 \end{bmatrix} \widehat{\mathbf{v}}_1 + \mathbf{v}_2 \\ &= \left\{ [0,\ I] + \widetilde{G}^{-1}_{k-1} \left[J^{(k)^T}_{12},\ I \right] G_k \begin{bmatrix} I & -J^{(k)}_{12} \\ 0 & 0 \end{bmatrix} \right\} \mathbf{v}. \end{aligned} \tag{4.4}$$

To summarize, for any given $\mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} \begin{matrix} \}\mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \}\mathcal{N}_{k-1} \end{matrix}$, we first compute $\widehat{\mathbf{v}}_1$ by (4.3) and then $\widehat{\mathbf{v}}_2$ by (4.4).

Conversely, for any given $\widehat{\mathbf{v}}_1$ and $\widehat{\mathbf{v}}_2$, we have from the first equation of (4.4) that

$$\mathbf{v}_2 = \widehat{\mathbf{v}}_2 - \widetilde{G}^{-1}_{k-1} I^{k-1}_k G_k \begin{bmatrix} I \\ 0 \end{bmatrix} \widehat{\mathbf{v}}_1 = [0,\ I](Y_1 \widehat{\mathbf{v}}_1 + Y_2 \widehat{\mathbf{v}}_2).$$

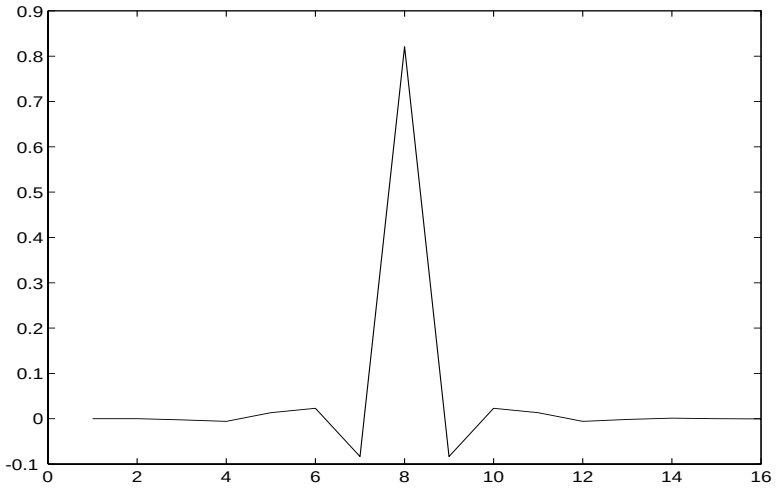

FIG. 12. *Cross-section plot of a wavelet-modified HB function; two Gauss–Seidel iterations.*

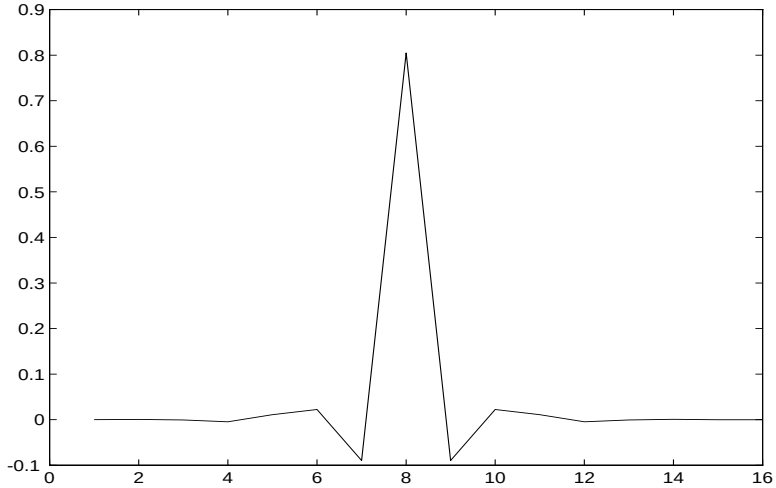

FIG. 13. *Cross-section plot of a wavelet-modified HB function;* 16 *CG iterations.*

It follows from (4.3) that $\mathbf{v}_1 = \widehat{\mathbf{v}}_1 + J_{12}^{(k)}\mathbf{v}_2 = [I,\ 0](Y_1\widehat{\mathbf{v}}_1 + Y_2\widehat{\mathbf{v}}_2)$, which verifies the identity (4.1) with $Y = [Y_1, Y_2]$ being the transformation matrix (defined in (4.2)).

We emphasize that the argument above provides algorithms for backward and forward actions of the transformation matrix $Y = [Y_1,\ Y_2]$. This is an important procedure in the implementation process.

Armed with the transformation matrix Y, we consider the problem

$$A^{(k)}\mathbf{v} = \mathbf{d}, \tag{4.5}$$

which is obtained by using the standard nodal basis. We transform it into the approximate wavelet-modified two-level HB by letting $\mathbf{v} = \mathbf{Y}\widehat{\mathbf{v}}$. In other words, we shall consider the transformed problem

$$Y^T A^{(k)} Y\widehat{\mathbf{v}} = \widehat{\mathbf{d}}, \quad \widehat{\mathbf{d}} = Y^T\mathbf{d}. \tag{4.6}$$

Using the two-level block partitioning of $\widehat{\mathbf{v}}$ and $Y = [Y_1,\ Y_2]$ we get the following two-by-two block system for the two-level AWM-HB components of $\widehat{\mathbf{v}}$:

$$\begin{bmatrix} \widehat{A}_{11}^{(k)} & \widehat{A}_{12}^{(k)} \\ \widehat{A}_{21}^{(k)} & \widehat{A}_{22}^{(k)} \end{bmatrix} \begin{bmatrix} \widehat{\mathbf{v}}_1 \\ \widehat{\mathbf{v}}_2 \end{bmatrix} = \begin{bmatrix} \widehat{\mathbf{d}}_1 \\ \widehat{\mathbf{d}}_2 \end{bmatrix}, \tag{4.7}$$

where

$$\begin{aligned} \widehat{A}_{11}^{(k)} &= {Y_1^{(k)}}^T A^{(k)} Y_1^{(k)}, \\ \widehat{A}_{12}^{(k)} &= {Y_1^{(k)}}^T A^{(k)} Y_2^{(k)}, \\ \widehat{A}_{21}^{(k)} &= {Y_2^{(k)}}^T A^{(k)} Y_1^{(k)}, \\ \widehat{A}_{22}^{(k)} &= {Y_2^{(k)}}^T A^{(k)} Y_2^{(k)} = I_k^{k-1} A^{(k)} I_{k-1}^k = A^{(k-1)}. \end{aligned}$$

Once $\widehat{\mathbf{v}}_1$ and $\widehat{\mathbf{v}}_2$ are known, the solution $\mathbf{v}$ of (4.5) can be recovered by using the formula (4.1). The transformed right-hand-side vector of (4.7) is given by

$$\begin{aligned} \widehat{\mathbf{d}}_1 &= {Y_1^{(k)}}^T \mathbf{d} = [I \quad 0] \left(I - G_k I_{k-1}^k \widetilde{G}_{k-1}^{-1} I_k^{k-1} \right) \mathbf{d}, \\ \widehat{\mathbf{d}}_2 &= {Y_2^{(k)}}^T \mathbf{d} = I_k^{k-1} \mathbf{d} = \left[{J_{12}^{(k)}}^T,\ I \right] \mathbf{d}. \end{aligned}$$

Therefore, the multiplicative AWM-HB preconditioner $B^{(k)}$ defined in section 2 takes the following block-matrix form:

$$\widehat{B}^{(k)} = \begin{bmatrix} \widehat{B}_{11}^{(k)} & 0 \\ \widehat{A}_{21}^{(k)} & B^{(k-1)} \end{bmatrix} \begin{bmatrix} I & {\widehat{B}_{11}^{(k)}}^{-1} \widehat{A}_{12}^{(k)} \\ 0 & I \end{bmatrix}. \tag{4.8}$$

Here $B^{(0)} = A^{(0)}$. The preconditioner $B^{(k)}$ is related to $\widehat{B}^{(k)}$ in the same way as $A^{(k)}$ to $\widehat{A}^{(k)}$; namely, $\widehat{B}^{(k)} = [Y_1, Y_2]^T B^{(k)} [Y_1, Y_2]$ and therefore, ${B^{(k)}}^{-1} = [Y_1, Y_2] {\widehat{B}^{(k)}}^{-1} [Y_1, Y_2]^T$. We will show below in Algorithm 4.1 that the inverse actions of $B^{(k)}$ can be computed only via the actions of $A^{(k)}$, Y_1, Y_2, and Y_1^T, Y_2^T in addition to the inverse actions of $\widehat{B}_{11}^{(k)}$.

We point out that (4.8) has precisely the same form as the algebraic multilevel method studied in Vassilevski [13], Axelsson and Vassilevski [2], and Vassilevski [14].

ALGORITHM 4.1 (computing inverse actions of $B^{(k)}$). *The inverse actions of $B^{(k)}$ are computed by solving the system*

$$B^{(k)} \mathbf{w} = \mathbf{d},$$

with the change of basis $\mathbf{w} = \mathbf{Y} \widehat{\mathbf{w}}$. Namely, by setting

$$\begin{aligned} \mathbf{w} &= Y_1 \widehat{\mathbf{w}}_1 + Y_2 \widehat{\mathbf{w}}_2 = [Y_1,\ Y_2] \begin{bmatrix} \widehat{\mathbf{w}}_1 \\ \widehat{\mathbf{w}}_2 \end{bmatrix}, \\ \widehat{\mathbf{d}}_1 &= Y_1^T \mathbf{d}, \\ \widehat{\mathbf{d}}_2 &= Y_2^T \mathbf{d}, \end{aligned}$$

$\mathbf{w} = {B^{(k)}}^{-1} \mathbf{d}$ is computed via the solution of $\widehat{B}^{(k)} \widehat{\mathbf{w}} = \widehat{\mathbf{d}}$ as follows:

- FORWARD RECURRENCE:
  1. *compute $\widehat{\mathbf{z}}_1 = {\widehat{B}_{11}^{(k)}}^{-1} \widehat{\mathbf{d}}_1$;*
  2. *change the basis; i.e., compute $\mathbf{z} = Y_1 \widehat{\mathbf{z}}_1$;*

3. *compute* $\widehat{\mathbf{d}}_2 := \widehat{\mathbf{d}}_2 - \widehat{A}_{21}^{(k)}\widehat{\mathbf{z}}_1 = Y_2^T(\mathbf{d} - A^{(k)}\mathbf{z})$;
4. *compute* $\widehat{\mathbf{w}}_2 = {B^{(k-1)}}^{-1}\widehat{\mathbf{d}}_2$;
5. *change the basis, i.e., compute* $\mathbf{v} = Y_2\widehat{\mathbf{w}}_2$;

- BACKWARD RECURRENCE:
  1. *update the fine–grid residual, i.e., compute*

$$\widehat{\mathbf{d}}_1 := \widehat{\mathbf{d}}_1 - \widehat{A}_{12}^{(k)}\widehat{\mathbf{w}}_2 = Y_1^T(\mathbf{d} - A^{(k)}Y_2\widehat{\mathbf{w}}_2) = Y_1^T(\mathbf{d} - A^{(k)}\mathbf{v});$$

  2. *compute* $\widehat{\mathbf{w}}_1 = {\widehat{B}_{11}^{(k)}}^{-1}\widehat{\mathbf{d}}_1$;
  3. *get the solution by* $\mathbf{w} = Y_1\widehat{\mathbf{w}}_1 + Y_2\widehat{\mathbf{w}}_2 = Y_1\widehat{\mathbf{w}}_1 + \mathbf{v}$.

END

Note that the above algorithm requires only the actions of the standard stiffness matrix $A^{(k)}$, the actions of the transformation matrices Y_1 and Y_2 and their transpositions Y_1^T and Y_2^T, and the inverse actions of $\widehat{B}_{11}^{(k)}$. The block ${\widehat{B}_{11}^{(k)}}^{-1}$ is some computationally feasible approximation to the well-conditioned matrix ${\widehat{A}_{11}^{(k)}}^{-1}$. The actions of Y^{-1} are not required in the algorithm.

We now formulate the solution procedure for one preconditioning step using the multiplicative AWM-HB preconditioner $B = B^{(J)}$.

ALGORITHM 4.2 (multiplicative AWM-HB preconditioning). *Given the problem*

$$B\mathbf{v} = \mathbf{d},$$

initiate

$$\mathbf{d}^{(J)} = \mathbf{d}.$$

Denote, for $k = 1, \ldots, J$,

$$\begin{aligned} Y_1^{(k)} &= \left(I - I_{k-1}^k \widetilde{G}_{k-1}^{-1} I_k^{k-1} G_k\right) \left[\begin{array}{c} I \\ 0 \end{array}\right] \begin{array}{l} \} \\ \} \end{array} \begin{array}{l} \mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \mathcal{N}_{k-1} \end{array}; \\ Y_2^{(k)} &= I_{k-1}^k. \end{aligned}$$

(A) **Forward recurrence:** *For* $k = J$ *down to* 1 *perform:*

1. *Compute:*

$$\widehat{\mathbf{d}}_1^{(k)} = {Y_1^{(k)}}^T \mathbf{d}^{(k)};$$

2. *Solve:*

$$\widehat{B}_{11}^{(k)}\widehat{\mathbf{w}}_1 = \widehat{\mathbf{d}}_1^{(k)};$$

3. *Transform basis:*

$$\mathbf{w} = Y_1^{(k)}\widehat{\mathbf{w}}_1;$$

4. *Coarse-grid defect restriction:*

$$\begin{aligned} \mathbf{d}^{(k-1)} &= {Y_2^{(k)}}^T \mathbf{d}^{(k)} - \widehat{A}_{21}^{(k)}\widehat{\mathbf{w}}_1 \\ &= {Y_2^{(k)}}^T(\mathbf{d}^{(k)} - A^{(k)}\mathbf{w}); \end{aligned}$$

5. *Set* $k = k - 1$. *If* $k > 0$ *go to* (1), *else*

6. *Solve on the* **coarsest level:**

$$A^{(0)}\mathbf{x}^{(0)} = \mathbf{d}^{(0)}.$$

(B) **Backward recurrence:**

1. *Interpolate result: Set* $k := k+1$ *and compute*

$$\mathbf{x}^{(k)} = Y_2^{(k)}\mathbf{x}^{(k-1)};$$

2. *Update fine-grid residual:*

$$\begin{aligned}\widehat{\mathbf{d}}_1^{(k)} &:= \widehat{\mathbf{d}}_1^{(k)} - \widehat{A}_{12}^{(k)}\mathbf{x}^{(k-1)}\\ &= \widehat{\mathbf{d}}_1^{(k)} - {Y_1^{(k)}}^T A^{(k)}\mathbf{x}^{(k)}\\ &= {Y_1^{(k)}}^T(\mathbf{d}^{(k)} - A^{(k)}\mathbf{x}^{(k)});\end{aligned}$$

3. *Solve:*

$$\widehat{B}_{11}^{(k)}\widehat{\mathbf{w}}_1 = \widehat{\mathbf{d}}_1^{(k)};$$

4. *Change the basis:*

$$\mathbf{w} = Y_1^{(k)}\widehat{\mathbf{w}}_1;$$

5. *Finally set*

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k)} + \mathbf{w};$$

6. *Set* $k := k+1$. *If* $k < J$ *go to step* (1) *of (B), else* **set**

$$\mathbf{v} = \mathbf{x}^{(J)}.$$

END

Similarly, one preconditioning solution step for the additive AWM-HB preconditioner $D = D^{(J)}$ takes the following form.

ALGORITHM 4.3 (additive AWM-HB preconditioning). *Given the problem*

$$D\mathbf{v} = \mathbf{d},$$

initiate:

$$\mathbf{d}^{(J)} = \mathbf{d}.$$

Denote, for $k = 1, \ldots, J$,

$$\begin{aligned}Y_1^{(k)} &= \left(I - I_{k-1}^k \widetilde{G}_{k-1}^{-1} I_k^{k-1} G_k\right)\begin{bmatrix} I \\ 0 \end{bmatrix}\ \begin{matrix}\}\\ \}\end{matrix}\ \begin{matrix}\mathcal{N}_k \setminus \mathcal{N}_{k-1}\\ \mathcal{N}_{k-1}\end{matrix},\\ Y_2^{(k)} &= I_{k-1}^k\end{aligned}$$

(A) **Forward recurrence:** *For* $k = J$ *down to* 1 *perform:*

1. *Compute:*

$$\widehat{\mathbf{d}}_1^{(k)} = {Y_1^{(k)}}^T\mathbf{d}^{(k)};$$

2. *Solve:*

$$\widehat{B}_{11}^{(k)}\widehat{\mathbf{w}}_1 = \widehat{\mathbf{d}}_1^{(k)};$$

3. *Transform basis:*

$$\mathbf{x}^{(k)} = Y_1^{(k)}\widehat{\mathbf{w}}_1;$$

4. *Coarse-grid defect restriction:*

$$\mathbf{d}^{(k-1)} = {Y_2^{(k)}}^T\mathbf{d}^{(k)};$$

5. *Set* $k = k-1$. *If* $k > 0$ *go to* (1), *else*
6. *Solve on the* **coarsest level:**

$$A^{(0)}\mathbf{x}^{(0)} = \mathbf{d}^{(0)}.$$

(B) **Backward recurrence:**

1. *Interpolate result: Set* $k := k+1$ *and compute*

$$\mathbf{w} = Y_2^{(k)}\mathbf{x}^{(k-1)};$$

2. *Update at level* k*:*

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k)} + \mathbf{w};$$

3. *Set* $k := k+1$. *If* $k < J$ *go to step* (1) *of (B), else* **set**

$$\mathbf{v} = \mathbf{x}^{(J)}.$$

END

For both the additive and multiplicative preconditioners, it is readily seen that the above implementations require only actions of the stiffness matrices $A^{(k)}$, the mass matrices G_k, and the transformation matrices I_{k-1}^k and I_k^{k-1}. The approximate inverse actions of $\widehat{A}_{11}^{(k)}$ can be computed via some inner iterative algorithms giving rise to the actions of $\widehat{B}_{11}^{(k)^{-1}}$. Similarly, the action of $\widetilde{G}_{k-1}^{-1}$ can be computed as approximate solutions of the corresponding mass-matrix problem using m steps of some simple iterative methods. Therefore, at each discretization level k, we perform a number of arithmetic operations proportional to the degrees of freedom at that level denoted by N_k. In the case of local mesh refinement, the corresponding operations involve only the stiffness and mass matrices computed for the subdomains where local refinement was made. Hence, even in the case of locally refined meshes, the cost of the AWM-HB methods is proportional to $N = N_J$. The proportionality constant depends linearly on $m = O(\log \tau^{-1})$ but is independent of J.

5. Numerical experiments.

5.1. 2D elliptic problems. The elliptic problem corresponds to the bilinear form given in (2.1), where the domain Ω is the unit square $(0,1)^2$. The finite element spaces V_k contain piecewise-linear continuous functions that vanish on $\Gamma_D \equiv \{(x,0) : 0 < x < 1\} \cup \{(0,y) : 0 < y < 1\}$. The spaces V_k correspond to uniform triangulations of Ω consisting of isosceles right triangles of size $h_k = 2^{-k}$ for $k = 0, 1, 2, \ldots, J$. The diffusion coefficient $a = a(x,y)$ in the bilinear form (2.1) was given by

$$a(x,y) = 1 + x^2 + y^2.$$

TABLE 1
HB additive preconditioners.

# levels J	$\lambda_{\min}$	$\lambda_{\max}$	κ	ρ	# iterations
3	0.193	2.161	11.16	0.458	27
4	0.130	2.520	19.35	0.578	39
5	0.095	2.790	29.29	0.648	50
6	0.076	2.997	39.00	0.695	60
7	0.062	3.158	50.41	0.730	70

TABLE 2
HB multiplicative preconditioners.

# levels J	$\lambda_{\min}$	$\lambda_{\max}$	κ	ρ	# iterations
3	0.380	1.000	2.627	0.159	11
4	0.290	1.000	3.447	0.264	15
5	0.226	1.000	4.422	0.322	18
6	0.181	1.000	5.519	0.362	20
7	0.148	1.000	6.724	0.397	23

The problem with $\widehat{A}_{11}^{(k)}$ was solved by the CG method. In other words, we may assume that the actions of $A_{11}^{(k)^{-1}}$ are exact within the machine accuracy. In the test, different numbers of inner iterations $m = 0, 1, 2, 4$ were applied to solve the mass-matrix problem in order to compute the actions of Q_{k-1}^a. The polynomial π_m in (3.4) was given by $\pi_m(t) = (1-t)^m$ and the following preconditioners are tested:

- *Jacobi:* In this case, the preconditioner B_{k-1} in (3.4) is the diagonal part of G_{k-1}.
- *Gauss–Seidel:* The preconditioner is given by $B_{k-1} = (D_{k-1} - L_{k-1})D_{k-1}^{-1}(D_{k-1} - U_{k-1})$ in (3.4). Here we have assumed the standard splitting $G_{k-1} = D_{k-1} - L_{k-1} - U_{k-1}$ of a matrix into diagonal, lower-triangular, and upper-triangular parts.

The Jacobi method gives rise to AWM-HB functions with local support, while the Gauss–Seidel method does not have this feature. But the computational cost for both methods is proportional to the total number of unknowns, since the actions of the inverse of $\widetilde{G}_{k-1}^{-1}$ are of optimal cost. Note that the standard HB method corresponds to $m = 0$ and $\pi_m(t) = 1$, which yields $\widetilde{G}_{k-1}^{-1} = 0$ in Algorithms 4.2 and 4.3. The multiplicative method with $m = 0$ then corresponds to the method of Vassilevski [13], which coincides with the HB-MG method of Bank, Dupont, and Yserentant [4]. The additive method with $m = 0$ is a variant of the HB method of Yserentant [18].

Tables 1–10 illustrate the number of iterations in the preconditioned conjugate gradient method applied to solving

$$A\mathbf{x} = \mathbf{b},$$

where $A = A^{(J)}$ for $J = 3, 4, 5, 6, 7$. The right-hand-side vector $\mathbf{b}$ was chosen to satisfy a prescribed solution $u(x, y)$.

The stopping criterion used is

$$\mathbf{r}^T W^{-1}\mathbf{r} \le 10^{-18}\mathbf{r}_0^T W^{-1}\mathbf{r}_0,$$

where W is the preconditioner B or D, $\mathbf{r}$ is the current residual, and $\mathbf{r}_0 = (I - AW^{-1})\mathbf{b}$ is the initial residual. We also show in Tables 1–10 the average convergence rate $\rho = \left[\sqrt{\frac{\mathbf{r}^T W^{-1}\mathbf{r}}{\mathbf{r}_0^T W^{-1}\mathbf{r}_0}}\right]^{\frac{1}{\text{iter}}}$.

TABLE 3
AWM-HB additive preconditioners; two Jacobi iterations.

# levels J	$\lambda_{\min}$	$\lambda_{\max}$	κ	ρ	# iterations
3	0.371	1.809	4.871	0.317	18
4	0.314	2.052	6.531	0.412	24
5	0.291	2.233	7.662	0.446	27
6	0.277	2.372	8.536	0.473	29
7	0.270	2.481	9.173	0.489	31

TABLE 4
AWM-HB multiplicative preconditioners; two Jacobi iterations.

# levels J	$\lambda_{\min}$	$\lambda_{\max}$	κ	ρ	# iterations
3	0.661	0.999	1.511	0.082	8
4	0.608	1.000	1.643	0.109	9
5	0.586	0.999	1.703	0.127	10
6	0.577	0.998	1.832	0.128	10
7	0.567	0.998	1.758	0.119	10

TABLE 5
AWM-HB additive preconditioners; four Jacobi iterations.

# levels J	$\lambda_{\min}$	$\lambda_{\max}$	κ	ρ	# iterations
3	0.354	1.832	5.175	0.336	19
4	0.302	2.071	6.855	0.419	24
5	0.285	2.250	7.885	0.454	27
6	0.278	2.386	8.566	0.474	29
7	0.275	2.492	9.055	0.485	30

TABLE 6
AWM-HB multiplicative preconditioners; four Jacobi iterations.

# levels J	$\lambda_{\min}$	$\lambda_{\max}$	κ	ρ	# iterations
3	0.640	1.000	1.561	0.086	8
4	0.589	1.000	1.742	0.121	10
5	0.569	0.999	1.808	0.135	10
6	0.563	0.998	1.856	0.137	11
7	0.563	0.998	1.905	0.137	11

TABLE 7
AWM-HB additive preconditioners; one Gauss–Seidel iteration.

# levels J	$\lambda_{\min}$	$\lambda_{\max}$	κ	ρ	# iterations
3	0.365	1.806	4.871	0.327	19
4	0.312	2.037	6.524	0.414	24
5	0.293	2.211	7.545	0.450	27
6	0.284	2.343	8.230	0.471	29
7	0.281	2.447	8.707	0.480	30

TABLE 8
AWM-HB multiplicative preconditioners; one Gauss–Seidel iteration.

# levels J	$\lambda_{\min}$	$\lambda_{\max}$	κ	ρ	# iterations
3	0.657	1.000	1.521	0.085	8
4	0.604	1.000	1.653	0.117	10
5	0.583	0.999	1.714	0.128	10
6	0.568	1.000	1.759	0.134	11
7	0.561	0.999	1.780	0.135	11

TABLE 9
AWM-HB additive preconditioners; two Gauss–Seidel iterations.

# levels J	$\lambda_{\min}$	$\lambda_{\max}$	κ	ρ	# iterations
3	0.353	1.829	5.178	0.344	20
4	0.298	2.067	6.919	0.423	25
5	0.279	2.247	8.037	0.457	28
6	0.272	2.384	8.753	0.480	30
7	0.269	2.491	9.235	0.490	31

TABLE 10
AWM-HB multiplicative preconditioners; two ($m = 2$) Gauss–Seidel iterations.

# levels J	$\lambda_{\min}$	$\lambda_{\max}$	κ	ρ	# iterations
3	0.639	1.000	1.564	0.086	8
4	0.584	1.000	1.710	0.118	10
5	0.561	1.000	1.782	0.137	11
6	0.553	0.999	1.805	0.138	11
7	0.550	0.999	1.816	0.141	11

Information on the minimum ($\lambda_{\min}$) and maximum ($\lambda_{\max}$) eigenvalues of $B^{(k)^{-1}}A^{(k)}$ and $D^{(k)^{-1}}A^{(k)}$ for $k = 3, \dots, J$ as well as the condition number $\kappa = \frac{\lambda_{\max}}{\lambda_{\min}}$ can also be found in Tables 1–10. The Lanczos method was employed in the code to provide this information.

The numerical experiments show a uniform convergence for the AWM-HB methods for relatively "large" $m = 3, 4$. This is well illustrated in Tables 6 and 10. The method is practically acceptable for a "small" iteration number $m = 1$ or $m = 2$. For example, the Jacobi method with $m = 2$ (see Tables 3, 4) and the Gauss–Seidel method with $m = 1$ (see Tables 7, 8) give weakly sensitive values on the number of iterations when J increases from 3 to 7. An improvement in terms of iteration counts over the standard HB method (see Tables 1, 2) is clearly demonstrated by this test.

To conclude, the numerical tests confirm the convergence theory presented in the first part of this work.

5.2. 3D elliptic problems. To assess the performance of the AWM-HB methods in a realistic situation, we present in this section some numerical results in 3D.

The test problem solves $u = u(x)$ satisfying

$$\mathcal{L}u \equiv -\sum_{i=1}^{3} \frac{\partial}{\partial x_i}\left(a_i(x_1, x_2, x_3)\frac{\partial u}{\partial x_i}\right) = f(x), \tag{5.1}$$

where $x = (x_1, x_2, x_3)$ and $\Omega = (0, 1)^3$. The Dirichlet boundary condition is imposed on (5.1). The coefficients a_i are given as follows:

$$\begin{aligned} a_1(x) &= 1 + 10\, x_1^2 + x_1^2 + x_3^2, \\ a_2(x) &= 1 + x_1^2 + 10\, x_2^2 + x_3^2, \\ a_3(x) &= 1 + x_1^2 + x_2^2 + 10\, x_3^2. \end{aligned}$$

The finite element partition of Ω is constructed in the following way. First, we partition Ω into small cubes of size $h_k = 2^{-k}$, $k = 0, 1, \dots, J$ for a given J. The vertices of the kth-level cubes form the nodes in $\mathcal{N}_k$. Second, each cube of level k with vertices $(x_1 + i_1 h,\ x_2 + i_2 h, x_3 + i_3 h)$, $i_1, i_2, i_3 = 0, 1$ is partitioned into six tetrahedrons

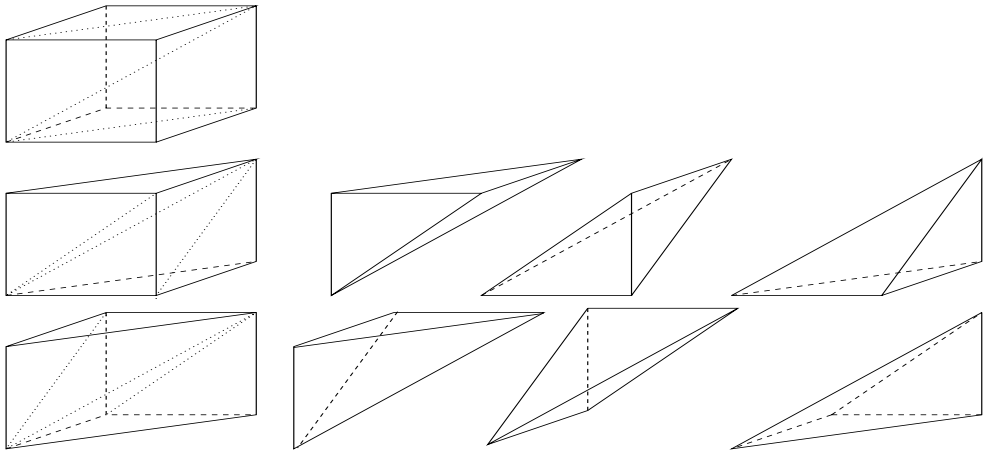

FIG. 14. *Cube partitioning into six tetrahedrons.*

by projecting its main diagonal connecting (x_1, x_2, x_3) with $(x_1+h_k, x_2+h_k, x_3+h_k)$ onto the six faces. These tetrahedrons form the kth-level triangulation $\mathcal{T}_k$. For the sake of clarity, the procedure is illustrated in Fig. 14. It can be seen that actually $\mathcal{T}_k$ is a refinement of $\mathcal{T}_{k-1}$.

The nodal interpolation operator giving rise to the matrix representation $I_{k-1}^k = \begin{bmatrix} J_{12}^{(k)} \\ I \end{bmatrix} \begin{matrix} \} \\ \} \end{matrix} \begin{matrix} \mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \mathcal{N}_{k-1} \end{matrix}$ can be implemented by a simple averaging procedure along the edges of the tetrahedrons. Algorithm 3.1 can be revisited for details.

Below we describe briefly how different methods are implemented in our numerical experiments.

HB-MG method. The implemented scheme is a version of the one presented in [4]. We are given $\widehat{A}^{(k)}$ in the two-level HB

$$\widehat{A}^{(k)} = \begin{bmatrix} A_{11}^{(k)} & \widehat{A}_{12}^{(k)} \\ \widehat{A}_{21} & A^{(k-1)} \end{bmatrix}. \tag{5.2}$$

Here $\widehat{A}_{12}^{(k)} = [I,\ 0]\, A^{(k)} I_{k-1}^k$. Starting with $B^{(0)} = A^{(0)}$, we define for $k = 1, 2, \ldots, J$,

$$\widehat{B}^{(k)} = \begin{bmatrix} D_{11}^{(k)} - L_{11}^{(k)} & 0 \\ \widehat{A}_{21}^{(k)} & I \end{bmatrix} \begin{bmatrix} D_{11}^{(k)^{-1}} & 0 \\ 0 & B^{(k-1)} \end{bmatrix} \begin{bmatrix} D_{11}^{(k)} - U_{11}^{(k)} & \widehat{A}_{12}^{(k)} \\ 0 & I \end{bmatrix} \begin{matrix} \} \\ \} \end{matrix} \begin{matrix} \mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \mathcal{N}_{k-1}. \end{matrix}$$

The inverse of the preconditioner is then given by

$$B^{(k)^{-1}} = \left[Y_1^{(k)},\ Y_2^{(k)}\right] \widehat{B}^{(k)^{-1}} \left[Y_1^{(k)},\ Y_2^{(k)}\right]^T,$$

where

$$\begin{aligned} Y_1^{(k)} &= \begin{bmatrix} I \\ 0 \end{bmatrix} \begin{matrix} \} \\ \} \end{matrix} \begin{matrix} \mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \mathcal{N}_{k-1} \end{matrix}, \\ Y_2^{(k)} &= I_{k-1}^k = \begin{bmatrix} J_{12}^{(k)} \\ I \end{bmatrix} \begin{matrix} \} \\ \} \end{matrix} \begin{matrix} \mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \mathcal{N}_{k-1} \end{matrix}. \end{aligned}$$

The blocks $D_{11}^{(k)}$, $L_{11}^{(k)}$, and $U_{11}^{(k)}$ come from the standard splitting $A_{11}^{(k)} = D_{11}^{(k)} - L_{11}^{(k)} - U_{11}^{(k)}$.

AWM-HB method. Starting with $B^{(0)} = A^{(0)}$, we recursively define for $k = 1, 2, \ldots, J$,

$$\widehat{B}^{(k)} = \begin{bmatrix} \widehat{B}_{11}^{(k)} & 0 \\ \widehat{A}_{21}^{(k)} & B^{(k-1)} \end{bmatrix} \begin{bmatrix} I & \widehat{B}_{11}^{(k)^{-1}} \widehat{A}_{12}^{(k)} \\ 0 & I \end{bmatrix} \begin{matrix} \} & (I - Q_{k-1}^a)(I_k - I_{k-1})V_k \\ \} & V_{k-1} \end{matrix},$$

and then let

$$B^{(k)^{-1}} = \left[Y_1^{(k)},\ Y_2^{(k)} \right] \widehat{B}^{(k)^{-1}} \left[Y_1^{(k)},\ Y_2^{(k)} \right]^T.$$

Here we have chosen

$$\widehat{B}_{11}^{(k)} \equiv B_{11}^{(k)} = (D_{11}^{(k)} - L_{11}^{(k)}) D_{11}^{(k)^{-1}} (D_{11}^{(k)} - U_{11}^{(k)}),$$

which stems from the symmetric Gauss–Seidel splitting of the first block $A_{11}^{(k)}$ in (5.2). Also, $\widehat{A}_{12}^{(k)} = Y_1^{(k)^T} A^{(k)} Y_2^{(k)}$ and the transformation matrices $Y_1^{(k)}$ and $Y_2^{(k)}$ are given by (see (4.2))

$$\begin{matrix} Y_2^{(k)} & = I_{k-1}^k, \\ Y_1^{(k)} & = \left(I - I_{k-1}^k \widetilde{G}_{k-1}^{-1} I_k^{k-1} G_k \right) \begin{bmatrix} I \\ 0 \end{bmatrix} \begin{matrix} \} & \mathcal{N}_k \setminus \mathcal{N}_{k-1} \\ \} & \mathcal{N}_{k-1} \end{matrix} \end{matrix}.$$

In the test presented below, the symmetric Gauss–Seidel method is further employed to obtain an approximation $\widetilde{G}_k$ to the Gram matrix G_k of level k. This formally corresponds to the choice $m = 1$ and $\pi_m(t) = 1 - t$ in the polynomial approximation (3.4).

MG method. Here is the method with one Gauss–Seidel smoothing. Starting with $B^{(0)} = A^{(0)}$, for $k = 1, 2, \ldots, J$, we recursively define

$$\widehat{B}^{(k)} = \begin{bmatrix} D^{(k)} - L^{(k)} & 0 \\ Y_2^{(k)^T} A^{(k)} & I \end{bmatrix} \begin{bmatrix} D^{(k)^{-1}} & 0 \\ 0 & B^{(k-1)} \end{bmatrix} \begin{bmatrix} D^{(k)} - U^{(k)} & A^{(k)} Y_2^{(k)} \\ 0 & I \end{bmatrix} \begin{matrix} \} & V_k \\ \} & V_{k-1} \end{matrix}$$

and then

$$B^{(k)^{-1}} = [I, Y_2^{(k)}] \widehat{B}^{(k)^{-1}} [I,\ Y_2^{(k)}]^T.$$

Here, $Y_2^{(k)} = I_{k-1}^k$ and we may formally set $Y_1^{(k)} = I$. Formally, we shall adopt the notation $\widehat{A}_{12}^{(k)} = Y_1^{(k)^T} A^{(k)} Y_2^{(k)} = A^{(k)} I_{k-1}^k$ and $\widehat{A}_{21}^{(k)} = Y_2^{(k)^T} A^{(k)} Y_1^{(k)} = I_k^{k-1} A^{(k)}$. This method has the same form as the HB-MG method with different transformation matrices $Y_1^{(k)}$ and $Y_2^{(k)}$ from the HB method. The major difference here is that $Y_1^{(k)}$ is a square matrix, and hence $\widehat{B}^{(k)}$ has a larger dimension than $B^{(k)}$.

The blocks $D^{(k)}$, $L^{(k)}$, and $U^{(k)}$ in $\widehat{B}^{(k)}$ correspond to the standard splitting $A^{(k)} = D^{(k)} - L^{(k)} - U^{(k)}$ into diagonal, lower-triangular and upper-triangular parts.

GS method. The preconditioner for $A^{(k)}$ is simply given by $B^{(k)} = (D^{(k)} - L^{(k)}) D^{(k)^{-1}} (D^{(k)} - U^{(k)})$.

The additive version corresponds to a simple deletion of the blocks $\widehat{A}_{12}^{(k)}$ and $\widehat{A}_{21}^{(k)}$, or $Y_2^{(k)^T} A^{(k)}$ and $A^{(k)} Y_2^{(k)}$ for the MG method. The additive form of the MG method is commonly referred to as the BPX method [6].

TABLE 11
Iteration counts for multiplicative methods.

	$h = 1/16$	$h = 1/32$	$h = 1/64$	$h = 1/128$
HB-MG	13	19	25	32
AWM-HB	10	10	10	10
MG	11	13	14	15
GS-PCG	11	24	51	105
# unknowns	3 375	29 791	250 047	2 048 383

TABLE 12
CPU timings (seconds) for multiplicative methods.

	$h = 1/16$	$h = 1/32$	$h = 1/64$	$h = 1/128$
HB-MG	4.72	57.0	601.79	6550.95
AWM-HB	9.66	81.51	656.80	5594.33
MG	2.28	22.74	195.81	1782.24
GS-PCG	1.13	20.37	342.49	5788.32
# unknowns	3 375	29 791	250 047	2 048 383

TABLE 13
CPU timings (seconds) for multiplicative methods with optimization.

	$h = 1/64$
HB-MG method	48.95
AWM-HB method	42.99
MG method	13.46
GS-PCG method	33.59
# unknowns	250 047

TABLE 14
Iteration counts for additive methods.

	$h = 1/16$	$h = 1/32$	$h = 1/64$	$h = 1/128$
HB	30	46	67	94
AWM-HB	16	18	19	20
BPX (MG & CG)	10	11	12	13
GS-PCG	11	24	51	105
# unknowns	3 375	29 791	250 047	2 048 383

TABLE 15
CPU-timings (seconds) for additive methods.

	$h = 1/16$	$h = 1/32$	$h = 1/64$	$h = 1/128$
HB-MG	5.82	72.68	853.96	1 0123.75
AWM-HB	7.57	69.73	598.62	5249.12
BPX (MG & CG)	2.10	18.94	165.55	1511.75
GS-PCG	1.11	19.99	341.91	5782.32
# unknowns	3 375	29 791	250 047	2 048 383

All methods are used as preconditioners in the CG iteration applied to $A\mathbf{x} = \mathbf{d}$, $A = A^{(J)}$, with the MG method as the only exception. The stopping criterion for a given preconditioner B was

$$\mathbf{r}^T B^{-1}\mathbf{r} \leq 10^{-8}\mathbf{r}_0^T B^{-1}\mathbf{r}_0,$$

where $\mathbf{r}$ is the current residual and $\mathbf{r}_0$ is the initial one.

The experiments were conducted by using a SUN Ultra 1 (170MHz) workstation.

The experiments (Tables 12 and 15) indicate that the MG method has the best performance in CPU timing. From Tables 11 and 14, we see that the convergence rate for the AWM-HB method is uniform, though the CPU timing is more than the MG method. We believe that there are other implementation methods which can improve the CPU timing significantly.

Finally, we remark that we should be careful when measuring CPU timings, since they depend on how the code is compiled. For example, we show in Table 13 the performance of the same methods as in Table 11, using the option *-fast* in the FORTRAN 77 compiler on a SUN Ultra 1 workstation. The speedup of the computation in the CPU timing is clearly significant for the test problems. The authors are indebted to Igor Kaporin for this observation.

REFERENCES

[1] O. Axelsson and I. Gustafsson, *Preconditioning and two–level multigrid methods of arbitrary degree of approximations*, Math. Comput., 40 (1983), pp. 219–242.

[2] O. Axelsson and P. S. Vassilevski, *Algebraic multilevel preconditioning methods*, II, SIAM J. Numer. Anal., 27 (1990), pp. 1569–1590.

[3] R. E. Bank and T. Dupont, *Analysis of a Two–Level Scheme for Solving Finite Element Equations*, Report CNA–159, Center for Numerical Analysis, The University of Texas at Austin, Austin, TX, 1980.

[4] R. E. Bank, T. Dupont, and H. Yserentant, *The hierarchical basis multigrid method*, Numer. Math., 52 (1988), pp. 427–458.

[5] F. Bornemann and H. Yserentant, *A basic norm equivalence for the theory of multilevel methods*, Numer. Math., 64 (1993), pp. 455–476.

[6] J. H. Bramble, J. E. Pasciak, and J. Xu, *Parallel multilevel preconditioners*, Math. Comput., 55 (1990), pp. 1–22.

[7] J. M. Carnicer, W. Dahmen, and J. M. Peña, *Local decompositions of refinable spaces and wavelets*, Appl. Comput. Harmon. Anal., 3 (1996), pp. 127–153.

[8] W. Dahmen and A. Kunoth, *Multilevel preconditioning*, Numer. Math., 63 (1992), pp. 315–344.

[9] M. Griebel and P. Oswald, *Tensor product type subspace splittings and multilevel iterative methods for anisotropic problems*, Adv. Comput. Math., 4 (1994), pp. 171–206.

[10] P. Oswald, *Multilevel Finite Element Approximation. Theory and Applications*, Teubner Skripten zur Numerik, Teubner, Stuttgart, 1994.

[11] R. Stevenson, *Robustness of the additive and multiplicative frequency decomposition multilevel method*, Computing, 54 (1995), pp. 331–346.

[12] R. Stevenson, *A Robust Hierarchical Basis Preconditioner on General Meshes*, Report 9533, Department of Mathematics, University of Nijmegen, Nijmegen, Netherlands, 1995.

[13] P. S. Vassilevski, *Nearly Optimal Iterative Methods for Solving Finite Element Elliptic Equations Based on the Multilevel Splitting of the Matrix*, Report 1989–09, Institute for Scientific Computation, University of Wyoming, Laramie, WY 1989.

[14] P. S. Vassilevski, *Hybrid V–cycle algebraic multilevel preconditioners*, Math. Comput., 58 (1992), pp. 489–512.

[15] P. S. Vassilevski, *On two ways of stabilizing the hierarchical basis multilevel methods*, SIAM Rev., 39 (1997), pp. 18–53.

[16] P. S. Vassilevski and J. Wang, *Stabilizing the hierarchical basis by approximate wavelets,* I: *Theory*, Numer. Linear Algebra Appl., 4 (1997), pp. 103–126.

[17] J. Xu, *Iterative methods by space decomposition and subspace correction*, SIAM Rev., 34 (1992), pp. 581–613.

[18] H. Yserentant, *On the multilevel splitting of finite element spaces*, Numer. Math., 49 (1986), pp. 379–412.

[19] H. Yserentant, *Old and new convergence proofs for multigrid algorithms*, Acta Numer., 1993, pp. 285–326.